CASCADE ERROR PROJECTION LEARNING THEORY

Tuan A. Duong*†, Allen R. Stubberud†, and Taher Daud*

* Center for Space Microelectronics Technology

Jet Propulsion Laboratory, California Institute of Technology

Pasadena, CA 91109

† Department of Electrical Engineering, University of California, Irvine

Irvine, CA 92717

Abstract:

The cascade correlation based neural network learning algorithm has drawn a lot of attention because of its enhanced learning capability. It overcomes significant drawbacks of error backpropagation (EBP) in that (1) it is no longer constrained to a fixed architecture via a preallocation of the number of hidden units, (2) it features selective weight training as opposed to EBP's global weight training. In addition, from a hardware implementation perspective, net works based on the cascade correlation algorithm require significantly less complex synaptic weight circuitry than those required by EBP.

We present a mathematical analysis for a new scheme termed Cascade Error Projection (CEP) and show tha((he same is also applicable to Cascade Correlation. In CEP, it is *shown that there exists, at least, a non zero set of weights which is calculated from affine space and that convergence is assured because the net work satisfies the Liapunov criteria, in added hidden units domain rather than* in *the time domain. The CEP technique is faster to execute because part of the weights are deterministically obtained, and the learning of weights from the inputs to each added hidden unit is performed as a single layer perceptron learning with previously obtained weights frozen. In addition, the initial weights start out with a zero value for every newly added unit, and a single hidden unit is applied instead of using a pool of candidate hidden units as for cascade correlation, Thereby,*

hardware implementation is also simplified. Furthermore, this analysis allows us to predict the suitability of other methods (such as the conjugate gradient descent and Newton's second order method) which will be good candidates for the learning technique. The choice of learning technique depends on the constraints of the problems (e.g., speed, performance, and hardware implementation); one technique may be nwre suitable than some of the others. Moreover, for a discrete weight space, the theoretical analysis presents the capability of learning with limited eight t quantization. Finally, 5- to 8-bit parity problems are investigated; the simulation results demonstrate that only three hidden units are required to learn the 5-bit parity problem with no error and the 6-bit parity problem with only one error, four hidden units are required to learn 7-bit parity problem with no error and8-bit problem with only one error. All simulations were done with a fixed 100 epoch iteration s for each single-layer perceptron (each single hidden unit) learning. In addition, with 3- to 4-bit weight resolution requirement it is demonstrated that this technique would be capable of learning reliably up to 8-bit parity problems.

I-Introduction

There are many ill-defined problems in pattern recognition, classification, vision, and speech recognition which need to be solved in real time [Duong et al. 1992, 1994, Boser et al. 1991]. These problems are too complex to be solved by a linear technique; the most suitable approach would be a non-linear technique, such as a neural network. An overview of general learning theory in neural network has been presented by Amari(1990), and a neural network survey has been summarized by Hecht-Nielsen(1 989). Currently, there are several neuromophic learning paradigms reported in literature [Cohen and Grossberg 1983, Rumelhart and McClelland 1986, Kosko 1988, Hinton et al. 1984, Albus 1971, Hopfield 1982, Widrow 1962, Fukushima 1982,

Rosenblatt 1958, Kohonen 1989, Jackson 1988, Falhman and Lebiere 1990, Duong et al. 1995] and a majority of them are supervised learning techniques. Error Backpropagation learning algorithm(EBP)[Rumelhardt and McClelland, 1986] is a very popular supervised learning technique. One of the most attractive features of the neural network is a massively parallel processing that offers tremendous speed only when implemented in hardware. From the hardware point of view, there are a few learning algorithms that are more practical than others [Tawel 1993]. Resource allocation [Platt 199 1], cascade correlation (CC) [Falhman and Lebiere 1990], and cascade backpropagation (CBP) [Duong et al. 1995] are among these practical learning algorithms. Generally, neural network approaches in hardware face two main obstacles: (1) difficulty of the network convergence due to the learning algorithm itself and the limited precision of the devices; (2) the very high cost of implementing hardware to truly mimic the synapse and neuron transfer functions dictated by the algorithm. Furthermore, the convergence and the implementable hardware have a mutual correlation to each other; for example, the convergence of the learning network depends on the weight resolution available in synapse [Hollis et al. 1990, Hoehfeld and Fahlman 1992], and the cost of implementation of each bit in synapse grows, at least doubly, in silicon area, power, and connectivity [Eberhardt et al 1989, Duong et al. 1992]. Avoidance of these obstacles requires a learning algorithm that has a reliable learning convergence (even with low weight quantization) and is simple to implement in VLSI hardware. There have been a few attempts to adapt such an algorithm to hardware. A weight perturbation [Jabri and Flower 1992, Alspector et al. 1993, Cauwenberghs 1993]

approach has been attempted. The technique of cascade correlation has shown encouraging results in learning via simulation; this method was shown to be reliable and fast in learning. However, it required at least a 12-bit weight resolution in synapse. A probabilistic technique [Hoehfeld and Fahlman 1992] has been introduced to augment the cascade correlation to reduce the weight resolution requirement, but it is difficult and the hardware implementation is complex by their proposed scheme. Another approach is the cascade backpropagation (CBP)[Duong et al.1995] that introduces a dynamical step size adjustment to overcome the hardware problem. In CBP approach, it was shown that the network is able to learn a relatively complex problem (6-bit parity) with only 5-bit weight-resolution synapse. However, learning, which is conducted in two layers by an iteration technique, is still slow and requites additional hardware when implemented.

Therefore, a better learning algorithm is required with a concrete mathematical foundation, as well as with a less costl y electronic implementation. In this paper, the cascade error projection (CEP) learning algorithm is presented. It offers a simple learning method using a one-layer perception approach and a deterministic calculation for the other layer. Such a simple procedure offers a very fast, reliable, and hardware implementable learning algorithm. In addition, the hardware implementation is not only tolerant of 3- and 4-bit weight resolution in synapse, but the learning technique which is robustly implementable in VLSI hardware is itself inherently simple. Furthermore, a detailed mathematical analysis of CEP is developed to guarantee the learning capability. To validate the new learning theory of CEP, simulations for 5- to 8-bit parity problems

are investigated with a variation in starting from a limited weight quantizations (3- to 6-bit weight resolution) and weight quantization of a floating point machine (32-bit for float and 64-bit for double precision).

In CC, the combination of supervised learning and cascading architecture shows that it is potentially better for implementation in hardware than EBP in terms of its learning capability [Hoehfeld and Fahlman 1992]. CBP is introduced to overcome the difficulty of EBP [Duong et al. 1995], with its gradient-descent technique of learning and independently training each layer. It allows one to dynamically change the stepsize of the synapse weight quantization to compensate for the low weight resolution synapse [Duong et al. 1995]. The cascade error projection learning algorithm is formed by adapting the cascading architecture from CC, the independent learning layer with a dynamical step size in CBP. In addition, a formal analysis is presented in the paper to integrate the three features.

The network architecture is shown in Figure 1. Shaded squares and circles indicate frozen weights; a square indicates calculated weights, and a circle indicates weights adapted by iterative learning procedure. The analysis will be based only on the set of weights that is connected to the new hidden unit (n+1), n being the number of hidden units already added. In this case, only the blank squares and circles will be require to be determined to improve the energy level.

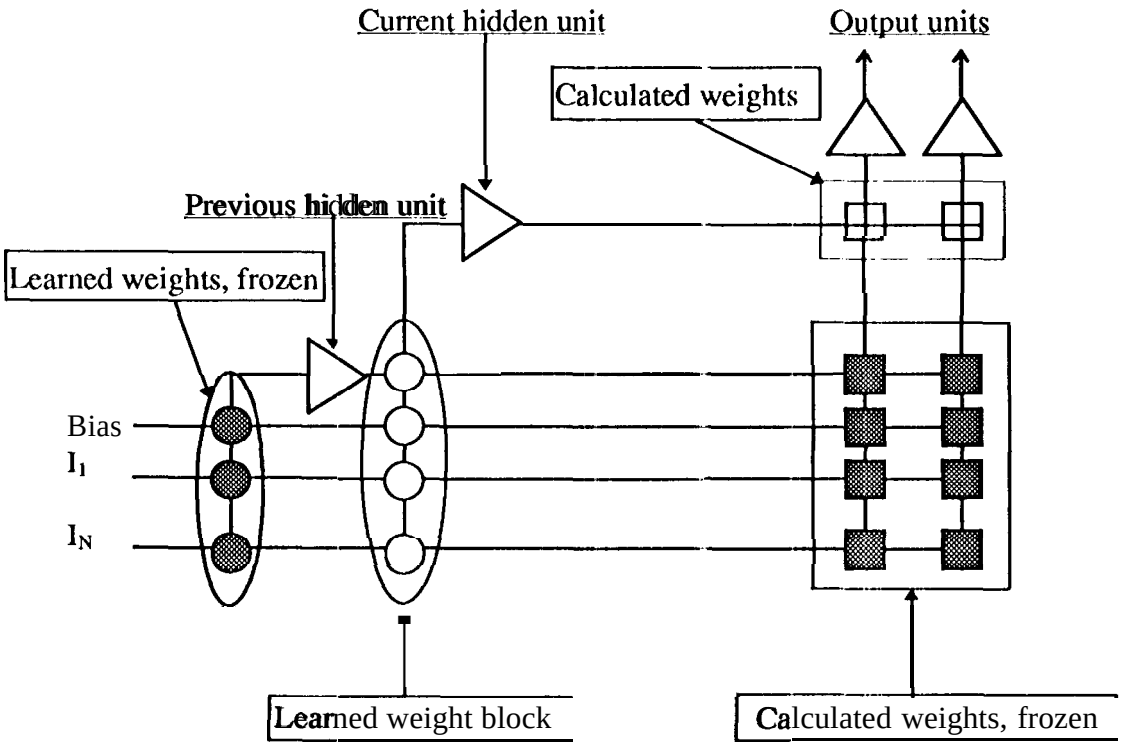

Figure 1. The architecture of cascade error projection includes inputs, hidden units, and output units. The shaded circles or squares indicate the learned or calculated weight set which arc computed and frozen. A circle indicates that learning is applied to obtain the weight set using perceptron learning, and a square indicates that the weight set is deterministically calculated.

This paper is designed as follows: Section 2 is an analysis of cascading architecture; a differential energy function between layer n and (n+1) is introduced. This function contains two sets of variables which are (1) the set of weights between the input (including previous expanded inputs) and the current hidden unit, namely W_{ih}; (2) the set of weights between the current hidden unit and the output unit, namely W_{ho}. The two sets of variables for the differential energy function are treated sequentially (not simultaneously). The differential energy function is maximized with respect to W_{ho} to

obtain $\max_{W_{ho}}(\Delta E)$. However, the $\max_{W_{ho}}(\Delta E)$ is a function of W_{ih}. In this section, we show that there exists a solution set W_{ih}^{*} which is obtained from affine space. This solution set W_{ih}^{*} guarantees that the network is reducing (or at least maintaining) the present energy level when the new hidden unit is added. In this case, we can conclude that the network converges in the Liapunov's sense. From this evidence, one can conclude that the solution which is obtained in non-linear space by Iearning techniques (gradient descent, conjugate gradient, correlation, or Newton's second order) may be the better one. This analysis ensures the convergence of learning in a non-linear space. In section 3, we present the convergence analysis as a function of synaptic resolution. It can be visualized that the learning trajectory is fairly smooth for high resolution synaptic weights, but is courser as the resolution is decreased, causing bigger jumps in the learning trajectory. Our analysis bears out that the necessary condition for the network to converge with limited weight resolution is the requirement of additional hidden units (compared with the number of hidden units in the floating point machine weight resolution). In section 4, the cascade error projection learning algorithm is proposed. This algorithm has two objectives: (a) consistency with the theoretical analysis and (b) easy implementation in silicon. Section 5 is used to validate the theory through the simulations. The problems that are used to validate the CEP are 5- to 8-bit parity problems in a floating point machine weight resolution, and limited weight resolution is studied. Discussions and suggestions are also included. Section 6 provides the conclusion of our analysis along with future research directions.

11. Mathematical Analysis in a Continuous Weight Space

Assume that the network contains n hidden units (see Fig. 2) and the learning cannot be improved any further in energy level. At this point, the new hidden unit (n+1) is added to the network.

a) Theory of Cascade Error Projection

Let Ξ be an input space and $\Xi \subset [-1,1]^{N+1}$, Y be an output space and $\Psi \subset [-1,1]^m$, and Ω be a hidden output space and $\Omega \subset [-1, I]^q$. Let us define

$$f_h(q):[-1,1]^{N+q} x \Re^{N+q} \longrightarrow [-1,1]$$

$$f_o :[-1,1]^{N+q+1} x \Re^{N+q+1} \longrightarrow y /$$

N is the dimension of the input space, q is the dimension of the expanded input space (q is dynamically changed and is based on the learning requirement), and m is the dimension of the output space. Finally, f is a sigmoidal transfer function which is defined:

$$f(x) = \frac{e^x - e^{-x}}{e^x + C\text{-}^{\prime\prime x}}$$

The used notations are defined as follows:

$\varepsilon_o^p = t_o^p - o_o^p$ (n) denotes the error of output element o and training pattern p with target t and actual output o(n). n indicates the output which has n hidden units in the network.

f'^{p}_{o} (n) = f'^{p}_{o} denotes the output transfer function derivative with respect to net_o of the output element o and the training pattern p.

$f_h^p(n+1)$ denotes the function of hidden unit n+1 and training pattern p.

X^p denotes the input pattern p and $|X|$ denotes the Euclidean distance of vector X.

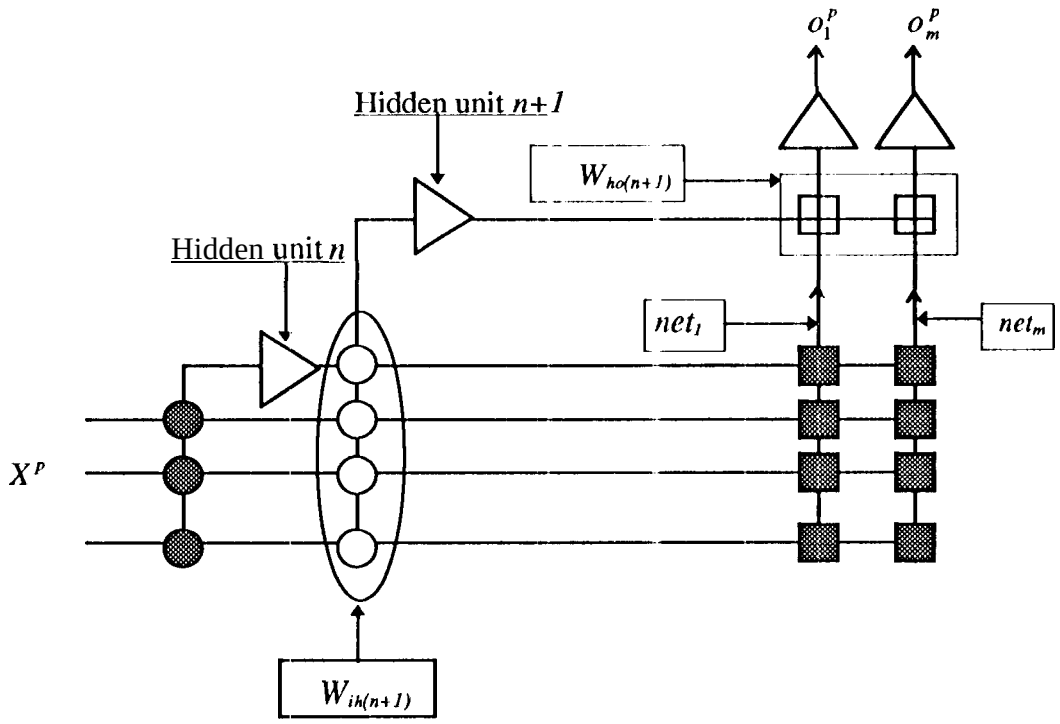

Figure 2: Assume that there are (n+1) hidden units in the network and the blank squares and circles arc the weight components which determine the weight values by learning or calculating.

Theorem 1: In cascading architecture, the maximum energy reduction between hidden unit n and (n+)) with respect to w_{ho} is

$$\sum_{p=1}^{P}\sum_{o=1}^{m}\varepsilon_o^p f'^p_o f_h^p(n+1)$$

Proof:

The energy function of the network can be defined as

$$E = \sum_{p=1}^{P} E^p = \sum_{p=1}^{P} \sum_{o=1}^{m} (t_o^p - o_o^p)^2 = \sum_{p=1}^{P} \sum_{o=1}^{m} (\varepsilon_o^p)^2$$

Let t_o^p be a target output of unit o and input pattern p, and the output of unit o is given:

$$o_o^p = f((X_i^p)^T W_{io} + \sum_{j=1}^{n+1} f_h^p(j) w_{ho}(j))$$

with

$$X_i^p = \begin{bmatrix} 1 \\ x_1^p \\ . \\ . \\ . \\ x_N^p \end{bmatrix} \quad ; \quad X_h^p(n+1) = \begin{bmatrix} f_h^p(1) \\ . \\ . \\ . \\ f_h^p(n+1) \end{bmatrix} ;$$

X_i^p (dimension $(N+1)x1$) denotes the original input vector of pattern p, and $X_h^p(n+1)$ (dimension $(n+1)x1$) denotes an expanded input vector with $(n+1)$ hidden units.

and let

$$i^p(n+1) = \begin{bmatrix} X : \\ X_h^p(n+1) \end{bmatrix}$$

then

$$f((i^p(j))^T W_{ih}(j)) = f_h^p(j+1)$$

$f_h^p(j+1)$ denotes the output of hidden unit $j+1$ with the input pattern p.

Let $E(n)$ and $E(n+1)$ be an energy level of the network with n, and $n+1$ hidden units respectively. The desire in learning is to reduce the energy from $E(n)$ to $E(n+1)$ as much as possible (ignoring the overlearning phenomenon). The ideal case would be

$$\max\{E(n) - E(n+-1)) = max \text{ AE}$$

From Appendix A, we have

$$\Delta E = \sum_{o=1}^{m}\{-w_{ho}^2 \sum_{p=1}^{P}[f'^{p}_{o} f_h^p(n+1)]^2 + 2w_{ho}\sum_{p=1}^{P}[\varepsilon_o^p f'^{p}_{o} f_h^p(n+1)]\} \qquad (1)$$

The sufficient condition for equation (1) to be maximum with respect to w_{ho} is

$$\max_{W_{ho}}(\Delta E) = \sum_{p=1}^{P}\sum_{o=1}^{m}\{\varepsilon_o^p f'^{p}_{o} f_h^p(n+1)\} \qquad \text{when } w_{ho} = \frac{\sum_{p=1}^{P}\varepsilon_o^p f'^{p}_{o} f_h^p(n+1)}{\sum_{p=1}^{P}[f'^{p}_{o} f_h^p(n+1)]^2} \qquad (2)$$

Theorem 2: There exists a subspace of weight space S c $\Re^{N+n}$ of which $W_{ih}(n+1) \in S$ has the new energy level $E(n+1)$, such that $E(n) - E(n+1) \geq 0$.

m

From equation (2), we have

$$\Delta E = (\sum_{p=1}^{P}\sum_{o=1}^{m} f_o'^{p} f_h^p(n+1)\{t_o^p - o_o^p\}$$

Randomly pick a set of weights, $W_{ih}(n+1)$, and validate all the training patterns. If $\Delta E \geq 0$, then the reduction energy of the network is satisfied, If not, change the polarity of W_{ih} (n +1), and since the transfer function is an odd function, ΔE will change in sign, which should satisfy the reduction energy.

Theorem 3: The projection of the error surface into the output of a new hidden unit always guarantees that there exists a weight subspace of the learning weight space,

which is obtained from affine space. These cascading sequential subspaces ensure that the network converges in Liapunov's sense.

Proof:

Let

$$\Gamma = \begin{bmatrix} \frac{1}{m}\sum_{o=1}^{m} f'^{1}_{o}\{t^{1}_{o} \quad o^{1}_{o}\} \\ \cdots\cdots\cdots\cdots \\ \cdots\cdots\cdots\cdots \\ \cdots\cdots\cdots\cdots \\ \frac{1}{m}\sum_{o=1}^{m} f'^{P}_{o}\{t^{P}_{o} - o^{P}_{o} \end{bmatrix} = \begin{bmatrix} \varphi_1 \\ \cdots \\ \cdots \\ \cdots \\ \varphi_P \end{bmatrix}$$

Then, $\Gamma \in [-1,1]^P$.

and

$$F_h(n+1) = \begin{bmatrix} f^{1}_{h}(n+1) \\ \cdots\cdots \\ \cdots\cdots \\ \cdots\cdots \\ f^{P}_{h}(n+1) \end{bmatrix}$$

We can rewrite equation (2) in a matrix form as follows:

$$AE = m\Gamma^{T} F_h(n+1)$$

let

$$F_h(n+1) = \Gamma \tag{3}$$

but

$$F_n(n+1) = F(IW_{ih}(n+1)) \tag{4}$$

with

$$I = \begin{bmatrix} (i'(n))^T \\ \cdots\cdots \\ \cdots\cdots \\ \cdots\cdots\cdots\ 1 \\ (i^P(n))^T \end{bmatrix}$$

From (3) and (4), let W_{ih}^* (n + 1) be a solution in affine space; then we have

$$IW_{ih}(n+1) = F_h^{-1}(\Gamma) \tag{5}$$

Finally, the solution is

$$W_{ih}^*(n+1) = I^+ F_h^{-1}(\Gamma)$$

I^+ is the pseudo-inverse of I [Haykin].

The existence of W_{ih}^* (n + 1) depends on the non-zero column matrix $I^+ F_h^{-1}(\Gamma)$, and the rank of I is at least 1 because of the non-linear combination of all previous dimensions $(i=1,n)$. At the same time, the error surface still exists (if it is zero, then the energy is already zero). Therefore, the existence of $W_{ih}^*(n+1)$ is almost always guaranteed. As shown, the existence in affine space is demonstrated; however, we are interested in non-linear space.

From *(3)* and (5), apply the mean value theorem:

$$\Gamma - F_h^*(n+1) = F'_h(C)\{F^{-1}(\Gamma) - IW_{ih}^*(n+1)\}$$

with

$$F'_h(C) = \begin{bmatrix} \text{“}f\text{‘}h(c^1)0\ldots0 \\ 0f'_h(c^2)\ldots0 \\ \ldots\ldots\ldots\ldots \\ 0\ldots0f'_h(c^P) \end{bmatrix} \quad \text{with } c^p \in \left((i^p)^T W_{ih}^*(n+1) \;,\; f^{-1}\{\frac{1}{m}\sum_{o=1}^{m} f'^{p}_{o}\varepsilon^{p}_{o}\} \right)$$

Note that the dimension of $F'_h(C)$ *is* PxP.

Let

$$\Gamma^* = F_h(IW_{ih}^*(n+1)) = \begin{bmatrix} \varphi_1^* \\ . \\ . \\ . \\ \varphi_P^* \end{bmatrix}$$

In other words, we have

$$r - r^* = F'_h(c)\{F'(r) - F'(r^*)\}$$

We also imply that

$$\Gamma - \Gamma^*|^2 = F'_h(C)\{F_h^{-1}(\Gamma) - F'_h(\Gamma^*)\}^2 \qquad (6)$$

Expanding equation (6), it is

$$r - r^{*2} = F'_h(C)F_h^{-1}(\Gamma)^2 - 2\{F'_h(C)F_h^{-1}(\Gamma)\}^T\{F'_h(C)F_h^{-1}(\Gamma^*)\} + F'_h(C)F_h^{-1}(\Gamma^*)^2 \quad (7)$$

Let

$$S = |\Gamma|^2 - |F'_h(C)F_h^{-1}(\Gamma)|^2 + 2\{F'_h(C)F_h^{-1}(\Gamma)\}^T\{F'_h(C)F_h^{-1}(\Gamma^*)\} - |F'_h(C)F_h^{-1}(\Gamma)|^2 \quad (8)$$

and $S \geq 0$ (9)

The inequality (9) is proved in Appendix B

From (8) and (9), it is obtained:

$$-2\Gamma^T\Gamma^* + r^{*2} \leq 0$$

In other words, it is

$$\Gamma^T F_h^*(n+1) \geq \frac{F_h^*(n+1)^2}{2}$$

and we also have

$$\Gamma^T F_h^*(n+1) \leq \frac{|\Gamma|^2 + |F_h^*(n+1)|^2}{2}$$

Finally, we have

$$\frac{F_h^*(n+1)^2}{2} \le \Gamma^T F_h^*(n+1) \le \frac{|\Gamma|^2 + F_h^*(n+1)|^2}{2} \quad (10)$$

We should note that $\frac{F_h^*(n+1)^2}{2} > 0$, because the rank of I is at least 1 (it is noted that the output of hidden unit n is non-linear combination of the all previous output hidden units and the original inputs and this output of hidden unit n ensures the rank of I at least 1 for itself). However, the inverse of sigmoidal function is used to obtain the $W_{ih}^*(n+1)$, so it is possible to encounter the null space in affine space. Therefore, the precise inequality is $\frac{F,*(n+1)2}{2} \ge 0$

From (1 O), there exists at least one solution obtained by pseudo-inverse technique in affine space. This solution also indicates the lower bound of reduction energy in the hidden unit (n+1). Therefore, in non-linear space it can be convinced there always exists a solution space when the error surface is projected to the new hidden unit for learning and the lower bound energy reduction is $m\frac{|F_h^*(n+1)|^2}{2}$. To obtain the maximum energy reduction, a straight approach is to obtain the most matching between $F_h(n+1)$ and 1", one can use gradient descent, maximum correlation, covariance, Newton's second order method, or conjugate gradient techniques.

Finally, $\nabla E(n) \le 0$, with $\nabla E(n) = E(n+1) - E(n)$.

In conclusion, we have shown that there exists a weight set $W_{ih}^*(n+1)$ which is obtained by the pseudo-inverse technique, and this weight set guarantees a reduction of

or remaining the same of energy with a new added hidden unit (n+1). From the network viewpoint, the energy decreases or remains the same when the number of hidden units increases; therefore the net work converges (in Liapunov's sense).

b) Discussion:

Up to this point, it is possible to discuss the learning algorithms by S. Fahlman (cascade correlation) and T. Duong (cascade error projection). In this analysis, it is clearly shown that the network converges when the error surface is projected to the current hidden unit. To find the weight set between the input units (original inputs and expanded inputs) and the current hidden unit depends on the objective which constraints to select the learning technique (e.g., if it is desired to have a software-based learning, then good candidates might be Newton's second or der, the maximum correlation or covariance technique, etc.) [Duong 1995]. As it is shown, the weight set can be obtained directly from affine space by using the pseudo-inverse technique, which has been thoroughly studied [Haykin 1991]. In addition, the mapping array technique also offers a well-defined methodology [Kung 1988] to solve a singular-value decomposition problem in the digital domain. However, in our approach, we are interested in a non-linear solution space in which the solution weight set can be obtained directly from a learning technique using analog/digital hardware. This learning approach will offer a better solution from both the theoretical and implementable point of view. First, the

solution which is obtained in non-linear space has compactness of the network and smoothness of the transformation because the data distribution is always in the non-linear domain. Second, it is hard to solve a singular-valued decomposition problem in a linear hardware network, even though the solution is deterministically defined, but the cost of the complicated hardware required by the network may exceed the available resources.

As stated in the arguments above, we believe the solution in non-linear space and the analog/digital hardware approach are more favorable to having an efficient, compact network and to providing simpler hardware implementation. Therefore, we have used theorem 3 as strong evidence to focus our attention on a learning technique to obtain the appropriate weight set for a particular use. The proposed learning algorithm CEP is more feasible for hardware (low quantization, fast learning, and simple design) [Duong 1995], Also from the theory, it is feasible to use the. conjugate gradient technique, or even better to use Newton's second order method to get a good software learning approach [Duong 1995]. However, the goal of our present analysis is to select the learning algorithm that best satisfies the given constraints.

III. Mathematical Analysis with Limited Weight Space

Theorem 4: In cascading architecture, the convergence of the network that is achieved in a high weight quantization space can also be obtained in limited weight quantization space (B is weight bit quantization available) such that:

$-\beta 2^{-(B+1)} \le \delta \le \beta 2^{-(B+1)}$ *and* $\Theta\{\delta\} = 0;$

The network with the limited weight quantization space is almost always converged in mean square sense which is

$$\Theta\{\tilde{F}_h^*(n+1)^2\} \le 2\Theta\{\Gamma^T \tilde{F}_h^*(n+1)\}$$

Θ is a statistical mean operator.

β is a dynamical coefficient.

Proof:

As proved above, the network converges in Liapunov's sense. In this section we also want to show that the CEP learning approach has the capability to learn in the discrete limited quantization space. The requirement for this learning capability is using the dynamical step size that can be obtained from the previous energy level. In Figure 2, with a new hidden unit $(n+1)$, the output o and pattern input p can be expressed:

$$o_o^p(n+1) = f(net_o^p + w_{ho} f_h^p(n+1)) \qquad (11)$$

In equation (11), w_{ho} is calculated; hence, it has very little effect on the learning capability, and it is ignored. However, The main focus of study in the learning capability of the network is W_{ih}. It can be expressed as follows:

$$W_{ih} = \tilde{W}_{ih} + \delta$$

and

$$\frac{-stepsize(n+1)}{2} \le \delta < \frac{stepsize(n+1)}{2} \qquad \text{with } stepsize(n+1) = \beta 2^{-B}$$

when $\tilde{W}_{ih}$ is a weight vector in discrete limited weight space, and δ is a noise vector that may come from the round-off technique.

We have

$$f_h^p(n+1) = f_h(i^p(\tilde{W}_{ih}(n+1) + \delta)) \qquad (12)$$

If δ is sufficiently small, equation (12) can be

$$f_h^p(n+1) = f_h(i^p\tilde{W}_{ih}) + f'_h(i^p\tilde{W}_{ih})i^p\delta$$

Let $\upsilon^p = f_h^p(n+1) - \tilde{f}_h^p(n+1)$ be an error between the hidden output with infinite weight resolution and the hidden output with limited weight resolution of hidden unit (n+1). Then,

$$\upsilon^p = f'_h(i^p\tilde{W}_{ih})i^p\delta$$

From the previous proof, we obtain

$$\frac{F_h^*(n+1)^2}{2} \le \Gamma^T F_h^*(n+1) \qquad (13)$$

Let $f'_h(i^p \tilde{W}_{ih}) = \tilde{f}'^p_h$

$$Y = \begin{bmatrix} \upsilon^1 \\ . \\ . = \\ . \\ \upsilon^P \end{bmatrix} \Rightarrow \begin{bmatrix} \tilde{f}'^1_h i^1 \delta \\ \cdots\cdots \\ . \; . \; ..\$.. \\ \cdots\cdots \\ \tilde{f}'^P_h i^P \delta \end{bmatrix}$$

From (13), we can rewrite

$$\left|\tilde{F}_h^*(n+1) + \mathrm{Y}\right|^2 \le 2\Gamma^T(\tilde{F}_h^*(n+1) + \mathrm{Y})$$

Expanded, we have

$$\left|\tilde{F}_h^*(n+1)\right|^2 + 2\{\mathrm{Y}^T \tilde{F}_h^*(n+1)\} + \left|\mathrm{Y}\right|^2 \le 2\{\Gamma^T \tilde{F}_h^*(n+1) + \Gamma^T \mathrm{Y}\} \qquad (14)$$

Let us introduce the *statistical mean operator* Θ. In the process of obtaining the weight set $\tilde{W}_{ih}$, the learning is repeatedly applied. This technique can be viewed as a statistical mean process, then (14) becomes

$$\Theta\{\tilde{F}_h^*(n+1)^2 + 2\{\mathrm{Y}^T \tilde{F}_h^*(n+1)\} + \left|\mathrm{Y}\right|^2\} \le 2\Theta\{\Gamma^T \tilde{F}_h^*(n+1) + \Gamma^T \mathrm{Y}\} \qquad (15)$$

But, Y is independent to 1“ and $F_h^*(n+1)$.

In the round-off technique, Y can reconsidered as white noise and $\Theta(Y)$ = O. Then inequality (15) becomes

$$\Theta\{|\tilde{F}_h^*(n+1)|^2\} \le 2\Gamma^T\Theta\{\tilde{F}_h^*(n+1)\} \qquad (16)$$

The result of inequality (16) guarantees the learning capability of the network if $F_h^*(n+1)$ *is* not zero, but it does not ensure the same achievement of energy level as does the infinite weight resolution. As analysis, inequality (16) only guarantees that the learning in limited weight quantization can be done, given the assumption $\delta << W_{ih}$. The remaining question is how small δ can be compared to W_{ih}, or how can we obtain information about δ through known information? We can observe that the smaller δ is, the closer the reduction between energies in limited weight quantization and infinite weight resolution is.

Discussion

The conversion between the continuous and the limited weight quantization weight space requires the scaling factor known as stepsize. With the fixed weight quantization levels (2” levels, and B is bit quantization), this stepsize is proportional to the energy

reduction level (ignoring the non linear factor which is come from non linear transfer function). The summarization can be described as follows:

$$\tilde{W}_{ih}(n+1) \propto stepsize(n+1)$$

and $\tilde{W}_{ih}(n+1) \propto F_h(n+1)$ (Roughly estimated and ignored the non linear factor)

$$F_h(n+1) \propto \Delta E(n)$$

$$F_h(n+1) \propto E(n)$$

then, $stepsize(n+1) \propto E(n)$

Therefore, $stepsize(n+1) = \alpha E(n)$ with α *constant*

IV. Cascade Error Projection Learning Algorithm

The motivation to use the cascade error projection technique is supported by three reasons:

1. It has fast learning by dividing the network into two independent networks:
    a) Stochastic learning network (Master network)
    b) Deterministic calculation network (Slave network).
2. It requires very low weight resolution which is easy to implement in hardware.
3. The learning algorithm theory guarantees the learning capability.

*Mathematical **approach:***

The energy function is defined:

$$E(n+1) = \sum_{p=1}^{P} \{f_h^p(n+1) - \frac{1}{m}\sum_{o=1}^{m}(t_o^p - o_o^p) f'^p_o\}^2$$

The weight updating between the inputs (including previous by expanded inputs) and the hidden unit is calculated as follows:

$$\Delta w_{ih}^p(n+1) = -\eta \frac{\partial E(n+1)}{\partial w_{ih}^p(n+1)} \qquad \text{(a)}$$

and the weight updating between hidden unit h and output unit o is

$$w_{ho}(n+1) = \frac{\sum_{p=1}^{P} \varepsilon_o^p f'^p_o f_h^p(n+1)}{\sum_{p=1}^{P} [f'^p_o f_h^p(n+1)]^2} \qquad \text{(b)}$$

*The Cascade Error Projection **Learning Algorithm** Procedure*

1. Start with the network which has input and output neurons. With the given input and output patterns and hyperbolic transfer function, onc can dctcrminc the set of weights between input and output by using pseudo-inverse or perceptron learning. The weight set Wio is thus obtained and frozen.
2. Add a ncw hidden unit with a zero weight set for each unit. In each loop (contains an epoch) an input-output pattern is picked up randomly in the epoch (no repeated pat tern until every pattern in the epoch is picked). Use

the perceptron learning technique of equation (a) to train $W_{ih}(n+1)$ for 100 loops.

3. Stop the perception training. Calculate the weights $W_{ho}(n+1)$ between the current hidden unit and the output units from equation (b).

   Cross-validating the network. If the criteria is satisfied, then stop training, and test the network. Otherwise, go to 2 until the number of hidden units is more than 20; then give up and quit!

V. Simulation

]n this section, we have selected complex non-linear problems for simulation which are 5- to 8-bit parity problems. In this simulation, there is only one output, therefore, we can speed up the network by ignoring the derivative of the output transfer function (the derivative of the output transfer function is always positive, therefore, the reduction of energy always holds) which is:

$$E(n+1) = \sum_{p=1}^{P} \{f_h^p(n+1) - s\varepsilon_o^p\}^2$$

s indicates the alternative sign (+/-) of the error surface to enhance the learning capability. From equation (2), with CEP learning algorithm the magnitude of the error ε_o^p is only a main object to minimize, but the sign of ε_o^p does not affect to the direction of energy reduction. The equation (2) can rewrite as follows:

$$\Delta E(n+1) = \sum_{p=1}^{P}\sum_{o=1}^{m}\{\varepsilon_o^p f'^p_o f_h^p(n+1)\} = \sum_{p=1}^{P}\sum_{o=1}^{m}\{f'^p_o(-\varepsilon^p_o)(f_h^p(n+1))\}$$

Therefore, the sign s that is used in this simulation does not affect the sign of reduction energy.

a) *ProblemLv*

The problems that are solved in this paper are 5- to 8-bit parity problems for(1) with no limited weight quantization (The weight resolution is the same as the floating point machine which is about 32-bit for floating point or 64-bit for double precision); and, (2) the limited weight quantization from 3-to 6-bits.

b) Parameters

As usual, the learning rate η is used and decrease linearly as follows:

$$\eta_{new} = \eta_{old} - .01 * \eta_0$$

In simulation, the parameter table below is used

	5-bit parity	6-bit parity	7-bit parity	8-bit parity
Floating-point machine Weight	$\eta_o = 1.0$ $\alpha = N/A$	$\eta_o = 1.0$ $a = N/A$	$\eta_o = 0.4$ $a = N/A$	$\eta_o = 0.4$ $a = N/A$
3-bit Weights	$\eta_o = 1.0$; $a = .0024810$	$\eta_o = 1.0$; $a = .016597$	$\eta_o = 1.0$; $a = .008766$	$\eta_o = 1.0$; $a = .004101$
4-bit Weights	$\eta_o = 1.0$; $a = .0016467$	$\eta_o = 1.0$; $a = .0108.58$	$\eta_o = 1.0$; $a = .008218$	$\eta_o = 1.0$; $a = .004101$
5-bit Weights	$\eta_o = 1.0$; $a = .0016467$	$\eta_o = 1.0$; $a = .010858$	$\eta_o = 1.0$; $a = .00816.3$	$\eta_o = 1.0$; $a = .004217$
6-bit Weights	$\eta_o = 1.0$;	$\eta_o = 1.0$;	$\eta_o = 1.0$;	$\eta_o = 1.0$;

$\alpha = .0016467$ $a = .010858$ $a = .008163$ $a = .004217$

c) Conversion technique (round-off technique)

The updating weight Aw is converted into the available weight quantization which is Aw*. The conversion can be summarized as follows:

$$stepsize(n) = \alpha E(n-1)$$

$$\Delta w^{*}_{jh}(n) = \begin{cases} stepsize(n) * \text{int}\left(\frac{\Delta w_{jh}}{stepsize(n)} + 0.5\right) & if \ \left(\frac{w_{jh}(n)}{stepsize(n)} + \text{int}\left(\frac{\Delta w_{jh}(n)}{stepsize(n)} + 0.5\right)\right) \le 2^{B} \text{ and } \Delta w_{jh}(n) > 0 \\ stepsize(n) * \text{int}\left(\frac{\Delta w_{jh}(n)}{stepsize(n)} - 0.5\right) & \text{if } \left(\frac{w_{jh}(n)}{stepsize(n)} + \text{int}\left(\frac{\Delta w_{jh}(n)}{stepsize(n)} - 0.5\right)\right) \le -2^{B} \text{ and } \Delta w_{jh}(n) < 0 \\ 0 & Otherwise \end{cases}$$

d) Simulation results:

We are using 5-,6-,7-, and 8-bit parity problems. The input and output highs are 0.8, and the lows are -0.8. The sigmoid is the hyperbolic tangent function with gain 1/2 ($f(x) = \frac{1-e^{-x}}{1+e^{x}}$). We use zero initial weights for each problem; therefore, we don't need to conduct a number of runs for each problem, Each hidden unit required 100 epoch learning iterations for the weight between the input and the current hidden unit only.

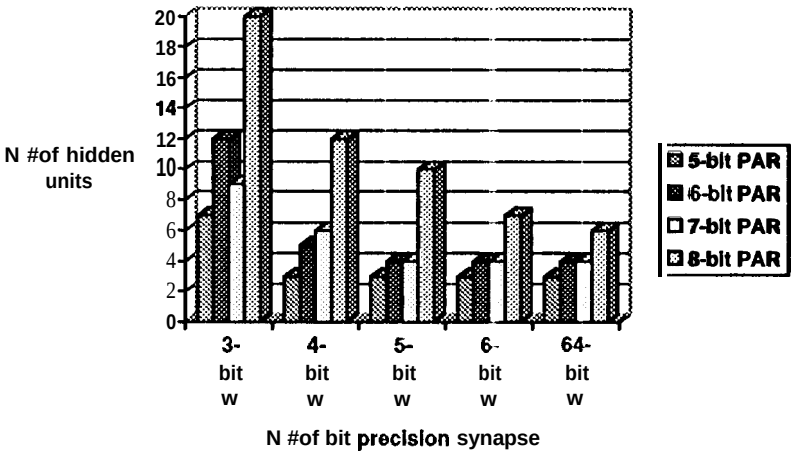

Figure 3: The chart shows CEP learning capability for 5- to 8-bit parity problems. x axis represents limited weight quantization (3-6 and 64-bit) and y axis shows the resulting number of hidden units (limited to 20). As shown, the lager number of hidden units compensate for the lower weight precision.

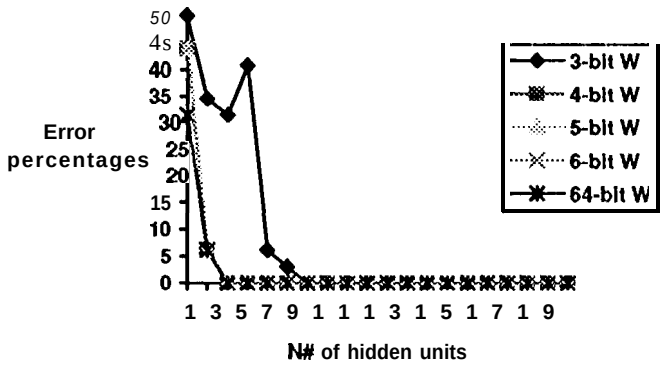

Figure 4: The figure shows the learning capability of CEP for 5-bit parity problem. x axis represents number of hidden units used (limited to 20), and y axis is percentage error. As shown, even with 3-bit limited weight quantization the network is able to learn 5-bit parity problem.

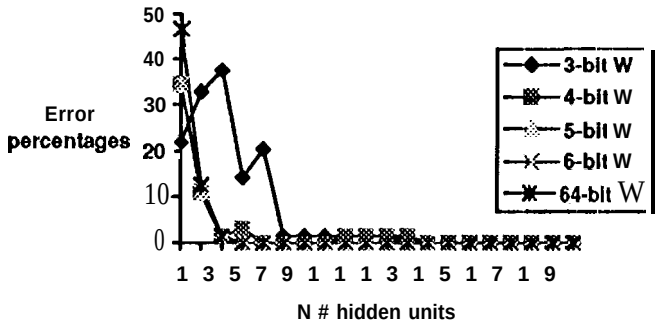

Figure 5: The figure shows the learning capability of CEP for 6-bit parity problem. x axis represents number of hidden units used (limited to 20), and y axis is percentage. As shown, even with 3-bit limited weight quantization the network is able to learn 6-bit parity problem.

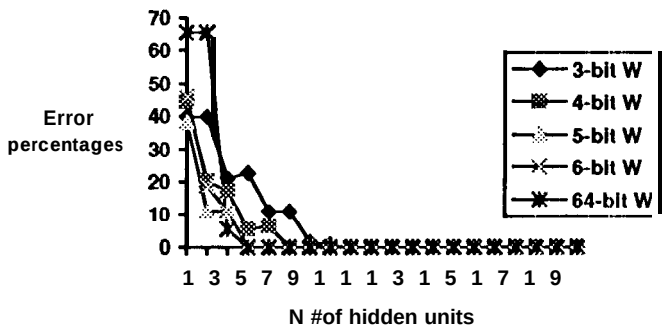

Figure 6: The figure shows the learning capability of CEP for 7-bit parity problem. x axis represents number of hidden units used (limited to 20), and y axis is percentage error. As shown, even with 3-bit limited weight quantization the network is able to learn 7-bit parity problem.

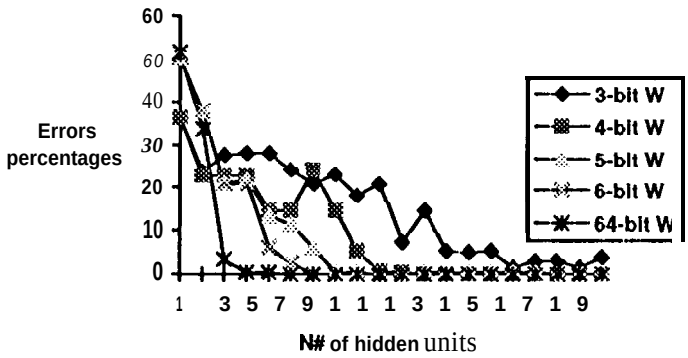

Figure 7: The figure shows the learning capability of CEP for 8-bit parity problem. x axis represents number of hidden units used (limited to 20), and y axis is percentage. As shown, with 3-bit limited weight quantization the network is unable to learn 8-bit parity problem perfectly, but the network is able to learn with 4-bit weight quantization perfectly.

VI. Discussion

As our mathematical analysis suggested, the weight set W_{ih} can be calculated deterministically by using the pseudo-inverse technique. However, instead of using the pseudo-inverse technique, perception learning is applied to obtain the following advantages and disadvantage:

+ A better weight solution set. The network solves the problem directly in non-linear space with non-linear measurements (input/output data set). In this approach, the problem is solved in a higher order space, instead of in linear space (first order) as used in the pseudo-inverse method.

+ A simpler hardware implementation required. With the perception learning method the analog or digital approach can be easily dorm It has a less stringent

hardware requirement as compared to that of the pseudo-inverse technique because perception learning technique is formulated less complicated to compare with the pseudo-inverse technique.

- Convergence problem. But the perception learning method faces more uncertainty in convergence than the pseudo-inverse technique. This is a trade-off between the non-linear and linear approach,

1) From the analysis point of view, there are many ways to obtain the weight set W_{ih} between the error surface and the output of a new hidden unit by using maximum correlation, covariant, conjugate gradient, and Newton's second order methods. One can investigate the most suitable technique for a particular application, as well as the resources available.

2) From Equation (2), we can prove that the neuron transfer function can be improved by using the non-monotonic function to expand the solution space. Or, to obtain a simpler mathematical model, a linear transformation can be used to eliminate the derivative of the output transfer function.

In the analysis of limited weight precision, the given assumption is $\delta << W_{ih}$. This assumption is very abstract, but it can vary, based on measured data distribution, Therefore, there is no determination of δ, but it is rather a relative perception of δ. To obtain δ in this paper, simulation is used.

VII. Conclusions

In this paper, we have shown that CEP is feasible for both a software- and a hardware-based learning algorithm. From this analysis, the way CC works can be understood in

depth. Moreover, the theoretical analysis provides us with the general framework of the learning architecture, and the particular learning algorithm can be independently studied for its suitability in a given application associated with some constraint for each problem. (For example, in the hardware approach CEP is most advantageous, and for software, Covariant or Newton's second order method is more advantages). For the CEP learning algorithm, the advantages can be summarized as follows:

. A fast and reliable learning technique

- An easy implementation in hardware
- A low weight resolution requirement in weight space

A robust model in learning neural networks

Future Research

Future research will be :(a) confirm each learning methods in simulation, (b) address the more practical problems to test the learning capability techniques as well as the hardware implementation requirements. We will study the signal-to-noise ratio (SNR), not only in the synapse, but for the entire network. Finally, we will propose a methodology of hardware implementation.

Acknowledgments:

The research described herein was performed by the Center for Space Microelectronics Technology, Jet Propulsion Laboratory, California Institute of Technology and was jointly sponsored by the Ballistic Missile Defense Organization/Innovative Science and Technology Office (BMDO/IST), the Office of Naval Research (ONR), the Advanced Research projects Agency (ARPA), and the National Aeronautics and Space Administration (NASA). The authors would like to thank Drs S. Eberdhardt, R. Tawel, and A. Thakoor for useful discussions, and Mr. P. Peerryman for a correct argument

References:

Albus J. S., "A theory of cerebellar function," *Mathematical Biosciences* vol. 10, pp. 25-61, 1971.

Alspector J. R., et al. "A parallel Gradient Descent Method for 1 earning in analog VLSI Neural Networks," In *Advances in Neural Information Processing Systems,* Vol. 5, pp. 836-844, Morgan Kaufman, San Mateo, CA 1993.

Amari S., "Mathematical foundations of neurocomputing," *Proc. IEEE, vol. 78,* NO. *9,* Sept, 1990.

Boser B. E., Sackinger E., Bromley J., LeCun Y., and Jackel L. D., "An Analog Neural Net work Processor with programmable Topology," *IEEE Journal of Solid State Circuits, vol. 26, NO. 12,* Dec. 1991.

Cauwenberghs G., "A Learning Analog Neural Network Chip with Continuous-Time Recurrent Dynamics," In Advanced in Neural Information Processing Systems, Vol. 6, pp. 858-865, Morgan Kaufman, San Mateo, CA 1993.

Cohen M. and Grossberg S., "Absolute stability of global pattern formation and parallel memmory stage by competitive neural networks, " *IEEE Trans. Systems, Man, Cybernetics,* vol. SMC-13, pp. 815-826, 1983.

Duong T. A., Brown T., Tran M., Langenbacher H., and Daud T., "Analog VLSI neural network building block chips for hardware-in-the-loop learning," *Proc. IEEE/INNS Int'l Join Conf. on Neural Networks,* Beijing, China, Nov. 3-6, 1992.

Duong T. A., Eberhardt S. P., Tran M. D., , Daud T., and Thakoor A. P., "Learning and Optimization with Cascaded VLSI Neural network Building-Block Chips," *Proc.*

IEEE/INNS international Join Conference on Neural Networks, June *7-11,1992,* Baltimore, MD, vol. I, pp. 184-189.

Duong T. A. et. al, "Low Power Analog Neurosynapse Chips for a 3-D "Sugarcube" Neuroprocessor," *Proc. of IEEE Intl' Conf. on Neural Networks*(ICNN/WCCI), Vol III, pp. 1907-1911, June 28-July 2, 1994, Orlando, Florida.

Duong T. A., Eberhardt S. P., Daud T., and Thakoor A., "Learning, in neural networks: VLSI implementation strategies," chapter of Handbook of Fuzzy logic and neural networks, Ed: C.H. Chen, Umass (to be published Sep. 1995).

Duong T. A., *manuscript in preparation,* 1995

Eberhardt S. P., Duong T. A., and Thakoor A. P., "Design of parallel hardware neural network systems from custom analog VLSI "building-block" chips," *IEEE/INNS Proc. IJCNN*, June 18-22, 1989 Washington D, C., vol. II, pp. 183.

Fahlmann S. E., Lebiere C., "The Cascade Correlation learning architecture," in *Advances in Neural Information Processing Systems II, Ed: D.* Touretzky, Morgan Kaufmann, San Mateo, CA, 1990, pp. 524-532.

Fukushima K., Miyake S., "Neocognitron: A new algorithm for pattern recognition telerant of deformations and shifts in position," *Pattern Recognition, 15 (6),* pp. 455-469, 1982.

Haykin S., *Adaptive Filter Theory,* Prentice-Hall, Inc. 2nd Ed., 1991.

Hecht-Nielsen R., *Neurocomputing,* Addison-Wesley Publishing Company, 1989.

Hinton G.F., Sejnowski T. J., and Ackley D.H., "Boltzmann machines: Constrain satisfaction networks that learn," *CMU Technical Report # CMU-CS-84-119,* Canergie Mellon University, Pittsburgh PA, May 1984.

Hoehfeld M. and Fahlman S., "Learning with limited numerical precision using the cascade-correlation algorithm," *IEEE Trans. Neural Networks,* vol.3, *No. 4,* pp 602-611, July 1992.

Hollis P. W., Harper J. S., and Paulos J. J., "The effects of Precision Constraints in a Backpropagation learning Network," *Neural Computation,* vol. 2, pp. 363-373, 1990.

Hopfield J. J., "Neural networks and physical systems with emergent collective computational abilities," *Proc. Natl. Acad. Sci. USA,* vol. *79,* pp. 2554-2558, 1982.

Jabri M. and Flower B. "Weight Perturbation: An Optimal Architecture and learning Technique for Analog VLSI Feedforward and recurrent Multilayer Networks," *IEEE Trans. on Neural Networks,* VO1 3, NO. 1, pp 154-157, Jan, 1992.

Jackson. I. R., "Convergence properties of radial basis functions", Constructive Approximation, vol. 4,243-264, 1988.

Kohonen T., "Self-Organization and Associative Memory," *Springer-Verlag,* Berlin Heidelberg,1989.

Kosko B., "Bidirectional associative memories," *IEEE Trans. Systems, Man and Cybernetics, vol.* 18, N# 1, pp. 49-60, 1988.

Kung S. Y., *VLSI Array Processors,* Prentice-Hall, Inc. l^{s}' Ed., 1988

Platt J., "A resource-allocating neural network for function interpolation," Neural Computation, vol. 3 (1991).

Rosenblatt F., "The perception: A probabilistic model for information storage and organization in the brain," *Psychology Review*, vol. 65, pp. 386-408, 1958.

Rumelhart D. E., and McClelland J. L., *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundation. MIT press, Cambridge, MA* 1986.

Tawel R., "Learning in analog neural network hardware," *Computers Elect. Engng*, Vol. 19, No. 6, pp 453-467, 1993.

Widrow B., "Generalization and information storage in networks of ADALINE neurons," Ed: G.T. Yovitt, "Self-Organizing Systems," *Spartan Books*, Washington DC, 1962.

Appendix A:

The energy function of the network can be defined as

$$E = \sum_{p=1}^{P} E^p = \sum_{p=1}^{P} \sum_{o=1}^{m} (t_o^p - o_o^p)^2$$

Assume that the network currently has n hidden units, and the energy no longer improves with any search techniques (gradient descent search, or exhausted search, etc.). The new hidden unit is now added to the network. The expected result is

$$E(n+1) \leq E(n)$$

This is equivalent to

$$\sum_{p=1}^{P} \sum_{o=1}^{m} \{t_o^p - f(net_o^p + w_{ho} f_h^p(n+1))\}^2 \leq \sum_{p=1}^{P} \sum_{o=1}^{m} \{t_o^p - f(net_o^p)\}^2$$

with $o_o^p = f(net_o^p)$

Expanding and rearranging, we have

$$\sum_{p=1}^{P}\sum_{o=1}^{m}\{[f(net_o^p + w_{ho}f_h^p(n+1)) - f(net_o^p)] \\ [f(net_o^p + w_{ho}f_h^p(n+1)) + f(net_o^p) - 2t_o^p]\} \le 0 \quad \text{(i)}$$

Assume that $w_{ho}f_h^p(n+1)$ is small so that

$$f\{net_o^p + w_{ho}f_h^p(n+1)\} \approx f(net_o^p) + f'(net_o^p)w_{ho}f_h^p(n+1) \quad \text{(ii)}$$

From (i) and (ii), it can be shown that

$$\sum_{p=1}^{P}\sum_{o=1}^{m}\{w_{ho}f'^p_o f_h^p(n+1)[w_{ho}f'^p_o f_h^p(n+1) - 2(t_o^p - o_o^p)]\} \le 0$$

with $f'(net_o^p) = f'^p_o$

or

$$\sum_{o=1}^{m}\{w_{ho}^2\sum_{p=1}^{P}[f'^p_o f_h^p(n+1)]^2 - 2w_{ho}\sum_{p=1}^{P}[f'^p_o f_h^p(n+1)\varepsilon_o^p]\} \le 0$$

Appendix B:

From equation (8), it is rewritten:

$$S = \sum_{i=1}^{P}[\varphi_i^2 - f'^2_h(c^i)\{f_h^{-1}(\varphi_i) - f_h^{-1}(\varphi_i^*)\}^2]$$

or

$$S = \sum_{i=1}^{P}[\{f'_h(d^i)f_h^{-1}(\varphi_i)\}^2 - f'^2_h(c^i)\{f_h^{-1}(\varphi_i) - f_h^{-1}(\varphi_i^*)\}^2]$$

a) if

$$\forall i \in \{1 \ldots P\}, \qquad \left|f_h^{-1}(\varphi_i)\right| \geq \left|f_h^{-1}(\varphi_i) - f_h^{-1}(\varphi_i^*)\right| \qquad \text{(iii)}$$

then

$$f'_{,}\ (d^i) \geq f'_{h}\ (c^i)$$

$$S \geq \sum_{i=1}^{P} f'^{2}_{h}(c^i)[\{f_h^{-1}(\varphi_i)\}^2 - \{f_h^{-1}(\varphi_i) - f_h^{-1}(\varphi_h^*)\}^2]$$

but

$$\forall i \in \{1 \ldots P\}, \qquad \{f_h^{-1}(\varphi_i)\}^2 - \{f_h^{-1}(\varphi_i) - f_h^{-1}(\varphi_h^*)\}^2 \geq 0$$

Hence,

$$S \geq 0$$

b)

If the assumption (iii) is no longer true, then vector Γ can be redefined as follows:

Let $|\varphi|_{\max} = \max\{|\varphi_i \qquad i=\{1 \ldots P\}$

and $\gamma_{\max} \cdot f^{-1}(|\varphi|_{\max})$

$$r = \begin{vmatrix} f(\frac{f^{-1}(\varphi_1)}{\gamma_{\max}}) \\ \cdots\cdots\cdots\cdots \\ \cdots\cdots\cdots\cdots \\ \cdots\cdots\cdots\cdots \\ f(\frac{f^{-1}(\varphi_I)}{\gamma_{\max}}) \end{vmatrix} = \begin{vmatrix} \phi_1 \\ \cdot \\ \cdot \\ \cdot \\ \phi_I \end{vmatrix}$$

and

$$\Gamma^* = F_h(IW^*_{ih}(n+1)) = \begin{bmatrix} \phi^*_1 \\ . \\ . \\ . \\ \phi^*_P \end{bmatrix}$$

with

$$\phi_i = \frac{e^{\gamma_i} - e^{-\gamma_i}}{e^{\gamma_i} + e^{-\gamma_i}}$$

γ^*_i is a component i^{th} of $F_h^{-1}(\Gamma^*)$ which is a pseudo-inverse solution of $F_h^{-1}(\Gamma)$

Obviously,

$\forall i \in \{1,\ldots P\}, \quad {}'y, \in [-\gamma_{\max}, \gamma_{\max}]$

and

$\forall i \in \{1,\ldots P\}, \quad \gamma^*_i \in [-\gamma_{\max}, \gamma_{\max}\,1$

ϕ_i and ϕ^*_i is expanding around zero and can be obtained as follows:

$$\phi_i = \gamma_i - \frac{1}{3}\gamma_i^3 + \xi$$

$$\phi^*_i = \gamma^*_i - \frac{1}{3}(\gamma^*_i)^3 + \xi^*$$

with $\xi, \xi^* \approx 0$

From equation (8), it can be reduced

$$S = |\Gamma|^2 - |\Gamma - \Gamma^*|^2 = \sum_{i=1}^{P} \phi_i^2 - (\phi_i - \phi^*_i)^2$$

Since magnitudes of $(\gamma_i)^3$ and $(\gamma_i^*)^3$ are very small to compare with the magnitude of γ_i and γ_i^* respective] y, and $F^{-1}(\Gamma^*)$ and $\{F^{-1}(\Gamma) - F^{-1}(\Gamma'^*)\}$ are orthogonal vectors [Kohonen 1989]. S can be simplified as below:

$$S = \sum_{i=1}^{P} \gamma_i^2 - (\gamma_i - \gamma_i^*)^2 = \sum_{i=1}^{P} (\gamma_i^*)^2$$

Since the approximation is used to simplify S, the sign of S is only interested for this proof.

Therefore, $S \geq 0$